⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　平2-196328

⑤1Int.Cl.[5]	識別記号	庁内整理番号	㊸公開　平成2年(1990)8月2日
G 06 F 7/52	310 C	7056-5B	

審査請求　有　請求項の数 1 (全13頁)

㊹発明の名称　浮動小数点演算装置

㉑特　願　平1-299887
㉒出　願　平1(1989)11月20日
優先権主張　㉜1989年1月13日㉝米国(US)㉛297016

㉒発明者　ロバート・ケビン・モントーイ　アメリカ合衆国テキサス州オウスチン、マウンテン・トライル6903番地
㉒発明者　ジョン・コーク　アメリカ合衆国ニューヨーク州ベツドフオード、バウント・リツヂ・ロード87番地
㉑出願人　インターナシヨナル・ビジネス・マシーンズ・コーポレーション　アメリカ合衆国10504、ニューヨーク州　アーモンク(番地なし)
㉔代理人　弁理士　頓宮　孝一　外1名

明　細　書

1．発明の名称　浮動小数点演算装置

2．特許請求の範囲

A×Bを乗算して第1部分結果を生成する手段、

Cを上記第1部分結果と桁合せする手段、

上記第1部分結果と上記の桁合せされたCを加え合わせる手段、

上記Cオペランドが上記第1部分結果同士の和よりも桁が高い場合に、上記Cオペランドを増分する手段、及び

上記結果を正規化する手段

を含み、

上記乗算がCオペランドの上記の桁合せと並行して実行される、

浮動小数点数演算(A×B+C)を実行するための装置。

3．発明の詳細な説明

A．産業上の利用分野

本発明は、一般にデータ処理に関し、より詳しくはA×B+C型の3元演算を浮動小数点数演算機構で実行する改良された装置に関する。

B．従来の技術

浮動小数点数計算の処理は、最新式コンピュータ演算にとって重要である。経験によれば、汎用演算処理装置は浮動小数点数の計算にあまり適していず、その結果、数値中心の計算を扱うために、専用の浮動小数点数演算機構(FPU)や演算処理装置が開発されている。

浮動小数点数演算用ハードウェアの潜在的ユーザは、デスクトップ・マイクロコンピュータから、信号処理システムや並列処理システム、さらには大型メインフレームにまで及んでいる。

浮動小数点数に対して加算、減算、乗算、除算など種々の演算を行なうのに、浮動小数点数演算機構が必要となることがある。浮動小数点用ハードウェアの中には、超越関数などその他の算術演算を支援する組込み機構を備えているものもある。

浮動小数点数演算処理装置がその機能を実行する速度を最大にすることは常に有用であるので、

性能利得を得るために用いられる既知の１つの方法は、特定の浮動小数点機能を実行する専用ハードウェアを設けることである。たとえば、算術関数のある種の組合せは、計算中で規則的に発生する。本発明は、Ａ×Ｂ＋Ｃ型の数式の計算に最適な、浮動小数点数演算処理装置で使用される装置を対象としている。様々な重要な数学的概念には、たとえば、

$$\sum_{i=1}^{3} A_i \times B_i = A_0 \times B_0 + A_1 \times B_1 + A_2 \times B_2 + A_3 \times B_3$$

の形の内積やＡ×³＋Ｂ×²＋Ｃｘ＋Ｄ＝Ｄ＋ｘ（Ｃ＋ｘ（Ｂ＋Ａｘ））というホーナー法など、この種の計算が含まれる。

多くの浮動小数点用ハードウェア機構は、ＶＬＳＩ（超大規模集積回路）を用いて実現され、ＶＬＳＩ浮動小数点数演算機構の設計者は、特定の機能が占める空間の大きさ、及び演算速度を最大にすることによる浮動小数点数演算機構の性能の最適化も考慮しなければならないことが多い。従来の浮動小数点数演算機構の設計では、乗算と加

算に別々のハードウェア機構を使用し、また乗加算（Ａ×Ｂ＋Ｃ）演算が頻繁に必要となるときは、上記の２つの機構を接続する方法を使用してきた。高速乗算には、IEEE Transactions on Computers、ＥＣ－１３、１９６４年２月、ｐｐ．１４～１７に所載の、Ｃ．Ｓ．ウォーレス（Wallace）の論文「高速乗算機構に関する提言（A Suggestion for a fast multiplier）」に示されているような高速加算機構がその最終段階で必要である。

高性能設計のためには、（Ａ×Ｂ＋Ｃ）を実行するハードウェアは下記のものを必要とする。

- ２個の加算機構（乗算用に１個と加算用に１個）
- ２個の丸め処理機構（乗算用に１個と加算用に１個）
- ４個の入力ポート（乗算用に２個と加算用に２個）
- ２個の出力ポート（乗算用に１個と加算用に１個）
- ２個の命令（乗算用に１個と加算用に１個）

本発明は、乗法演算子と加法演算子を組み合わ

せることにより必要な要素を減少させるものである。

Ｃ．発明が解決しようとする課題

したがって、本発明の目的は、Ａ×Ｂ＋Ｃ（Ａ、Ｂ、Ｃは浮動小数点数）の演算を行なえる単一ハードウェア構造を提供することである。

もう一つの目的は、入力からＡ×Ｂ＋Ｃ演算の結果までの遅延を最小にすることである。

もう一つの目的は、１回の丸め演算を行なうことにより、Ａ×Ｂ＋Ｃ演算の精度を上げることにある。

もう一つの目的は、Ａ×Ｂ＋Ｃ演算を行なうのに１つの機構だけで済ませることにより、必要なハードウェアを減らすことである。

もう一つの目的は、Ａ×Ｂ＋Ｃ演算用の３つの入力ポートと１つの出力ポートを備えた単一の機構を作成することにより、入力／出力ポートが少なくなった機構を提供することである。

もう一つの目的は、Ａ×Ｂ＋Ｃ演算を表現するための３つの入力オペランドと１つの出力オペラ

ンドを備えた機構を作成することにより、命令要件が減少した機構を提供することである。

Ｄ．課題を解決するための手段

本発明の目的及び特徴を説明する好ましいが例示的な実施例によれば、Ａ×Ｂ＋Ｃ型の浮動小数点数演算を実行するための新しい装置と方法が提供される。ＡとＢの乗算を実行し、それと同時に加数Ｃを加算のために桁合せさせる。

結果Ａ×Ｂ＋Ｃ（Ａ、Ｂ、Ｃは浮動小数点数）を生成する単一の浮動小数点数演算機構が提供される。オペランドＣは、乗算の開始段階と並行してシフトされる。結果は、１回の加算と正規化によって生成され、ハードウェア、遅延、及び丸めの誤差が減少する。

Ｅ．実施例

本発明は、Ａ×Ｂ＋Ｃ型の高速かつ正確な浮動小数点数算術演算を実行する装置を提供する。

浮動小数点数は、符号付き仮数に基数の整数べきをかけた形をとる。すなわち、１０進表記法では、数１０１．３２は０．１０１３２×１０³と

書かれ、3が指数、0．10132が仮数である。この例で、数の基数または基底は10である。浮動小数点数表記法は、またその他の基底を用いた数にも使用でき、高速ディジタル・コンピュータの場合には、浮動小数点数は2進表示である。したがって、101．011の形の2進数は、0．101011×2^3の形の浮動小数点数として書くことができ、その仮数は0．101011、指数は3、基数または基底は10であり、点は10進小数点でなく2進小数点と呼ばれる。もちろん、ディジタル・コンピュータでは、指数3は2進数11となる。

2進浮動小数点数の加算を行なう場合、加算を正しく行なうには、両方の数を2進小数点に関して桁合せさせなければならないことが分かる。加算を実行する場合、加え合わせる両方の数が同じ指数をもたなければならない。その後は、仮数をそのまま加えることができる。

乗算では、いくつかの既知の技法のどれかを使って仮数を掛け合わせ、指数を加え合わせる。それ

ぞれMビット幅及びNビット幅の仮数を有するAとBを掛け合わせる場合、結果の最大長がM＋Nであることは明らかである。指数は両方の指数の加算によって生じる大きさになる。また、A×Bの結果に加えようとする数Cがこの結果と同じ指数を持たない可能性が大きく、したがってA×Bの結果と正しく桁合せされるように数Cをシフトしなければならないことは明らかである。

本発明は、A×B＋C型の演算を行なうものである。単純な乗算A×BはC＝0とおくことによって実行でき、A＋Cという単純な加算はB（またはA）＝1とおくことによって実行できるので、このような機構は、論理演算機構（ALU）の基礎として使用できることが理解できるはずである。

A×B＋C（A、B、Cはmビットの仮数とeビットの指数をもつ浮動小数点数）の演算を考える。本発明では、Cオペランドは、CオペランドをAの指数＋Bの指数－Cの指数に等しいビット数だけシフトすることにより、AとBの浮動小数点積と桁合せされる。本発明では、この動作は、

乗算で必要なビット生成・圧縮と並行して行なうことができる。部分乗数を使って、和がA×Bの結果に等しい2つの加数を得る。これらの加数、すなわち部分積は、Cオペランドのシフトと並行して求められる。

部分積を、最終積を得るために加え合わさなければならない2つの数に簡約するために、乗算に、少なくともlog（m）（ただし、mは入力ワードのビット数）の時間がかかることは周知である。この乗算時間中にC項をA×B演算の積と桁合せさせることにより、加算は乗算にほとんど遅延を追加しない。C項を桁合せし簡約した後、この2つの項の最終加算を行なわなければならない。Cの指数がAとBの指数の和より2m＋1ビット以上小さい場合は、Cの結果は、AとBの乗算におけるどのビットよりも桁が低い。したがって、Cのビットは、A×Bの範囲から「シフトアウトされ」、積に使用されない。A×B＋Cの演算で、Cの指数がAとBの指数の和よりわずかな（m未満の）値だけ大きい場合には、乗算の完成に必要

な加算からオーバーフローが生じる可能性がある。このオーバーフローは、繰上げのある場合に入力を増分する加算機構として機能する増分機構中のCシフト機構のオーバーフロー範囲に加えなければならない。

Cの指数がAの指数とBの指数の和よりもm＋1以上大きい場合、乗加算演算の結果はCである。Cの指数がAの指数とBの指数の和よりも2m＋1以上小さい場合は、乗加算演算の結果はCである。指数の差が3mを越える場合は、結果はC（Cの指数の方が大きい場合）またはA×Bとなる。したがって、（乗算に必要な）2mビットの加算機構及び（オーバーフロー範囲に必要な）mビットの増分機構を使って、最終結果を生成しなければならない。次いで、先行ゼロを除去し、最大の精度をあげるため、3mの結果を正規化し丸めなければならない。

次に、本発明の好ましい実施例の構成図を示す第1図を参照する。指数演算機構10は、3つの指数EXP（A）、EXP（B）、EXP（C）

を受け取る。指数演算機構１０の主要機能は、ＥＸＰ（Ａ）＋ＥＸＰ（Ｂ）－ＥＸＰ（Ｃ）の値を求めることであり、これは加算機構によって行なわれる。指数演算機構１０は、符号付き数の処理などに関連する追加機能を有する。本発明は、符号ビットを有する符号付き数を使用することを意図するものである。ただし、０の符号ビットは正数を示し、１の符号ビットは負数を示す。符号ビットは、数の内部でのその使用が首尾一貫している限り、様々な場所に置くことができる。最も普通のシステムでは、符号ビットは最上位ビットの位置を占める。

符号付き数は、それ自体の補数形に変換すると、ディジタル回路で容易に処理できる。本発明では、Ａ、Ｂ、Ｃの符号が指数演算機構１０中で比較される。Ｃの符号がＡ×Ｂの結果と異なっていると比較機構１１で判定された場合、シフト機構１４の出力は（オーバーフローも含めて）、補数化機構１５によって１の補数の形に補数化される。補数化機構１５は、第２図に示すように構成するこ

とができ、排他的ＯＲゲート４０及び４１を含んでいる。当業者にとって明白なように、排他的ＯＲゲートの数は、システム中で使用される２進数のビット数に依存する。補数信号を端子１５Ａで受け取ったときは常に、ＤＡＴＡ　ＩＮが補数化され、ＤＡＴＡ　ＯＵＴとして供給される。

それぞれＭＡＮ（Ａ）及びＭＡＮ（Ｂ）で表わされる、Ａ及びＢの仮数を部分乗算機構１２が受け取る。部分乗算機構１２の動作についてはさらにあとで説明する。部分乗算機構１２は、ＡとＢを掛け合わせるが、和がＡ×Ｂである２つの加数から構成される部分積だけを与える。

ＭＡＮ（Ｃ）で表わされるオペランドＣの仮数は、シフト機構１４に供給される。シフト機構１４は、通常のシフト機構の方式で動作して、ＣをＥＸＰ（Ａ）＋ＥＸＰ（Ｂ）－ＥＸＰ（Ｃ）の計算から求められる量だけ右へシフトする。この値がシフト機構１４の入力側１４Ａに供給され、シフト機構がその入力ＭＡＮ（Ｃ）を左へシフトする量を制御する。Ｃ shifted で表わされるＭＡ

Ｎ（Ｃ）のシフトされた出力が、部分乗算機構１２からの部分積と共に、繰上げ／保管加算機構１６に供給される。負のシフト演算（ＥＸＰ（Ａ）＋ＥＸＰ（Ｂ）－ＥＸＰ（Ｃ））からのオーバーフローがある場合は、左シフトが行なわれる。ＣがＡ及びＢより桁が高い、すなわちＥＸＰ（Ｃ）＞ＥＸＰ（Ａ）＋ＥＸＰ（Ｂ）のときは常に、オーバーフローが生じることに留意されたい。

繰上げ／保管加算機構１６は、３つの入力と２つの出力をもつ、当技術分野で周知の通常の繰上げ／保管加算機構である。２つの出力とは和及び繰上げ出力であり、それぞれＳ及びＣで表わされる。

繰上げ／保管加算機構１６のＣ出力及びＳ出力は全加算機構１８に供給される。全加算機構１８は、繰上げ／保管加算機構１６からのＣとＳの２つの結果を加え合わせる、当技術分野で周知の通常の加算機構である。全加算機構１８はまた、キャリー・イン（下位からの繰上り）を受け取るキャリー・イン（ＣＩ）入力ポート、及び加法演算の

結果実際にキャリー・アウト（上位への繰上げ）が生じる場合にキャリー・アウトを出すキャリー・アウト（ＣＯ）出力ポートを備えている。

比較機構１１からの信号も、リード線１７を介して増分機構２０に１の補数符号として供給され、第１ビット位置に置かれる。次いで、この信号は、増分機構２０による増分の結果に応じて、最終的に補数化機構２２の端子２２Ａに転送され、必要に応じて、補数化機構２２での補数化をオンにしたりオフにしたりする。

ＣＩは増分機構２０から受け取られる。増分機構２０はシフト演算機構１４からオーバーフローを受け取る。増分機構２０は、１つの入力をゼロに設定すると、加算機構として機能する。すなわち、全加算機構１８からのＣＯがあり、このＣＯが加算機構２０のキャリー・イン（ＣＩ）入力ポートに供給される場合に、シフト機構１４からのオーバーフローを増分する働きをする。増分機構２０での増分の結果がキャリー・アウト（ＣＯ）をもたらす場合には、このＣＯが全加算機構１８の上

記ＣＩ入力ポートに供給される。増分された出力は２０Ａに供給される。

補数化機構２２は、全加算機構１８と増分機構２０の出力を受け取り、受け取った値を補数化する。これは、上記のように符号付き数を処理するために必要である。

正規化機構２４は、先行ゼロを除去し、結果の精度を最大にする働きをする。正規化機構２４は、先行ゼロを認識し、仮数をシフトして、それに応じて指数を増分または減分する働きをする回路なら、どれによっても実現できる。この演算を実行する特に高速の１つの回路は、１９８８年１０月７日付けで出願され、本出願人に譲渡された、「先行０／１予測機構（Leading 0/1 Anticipator (LZA)」と題する関連米国特許出願第２５５０８９号に記載されている。この回路を用いると、結果を求める前に先行ゼロの決定が可能となり、したがって遅延が追加されることはない。

乗法加法演算の桁数を必要な精度、多くは入力

の原精度に一致させるために丸めが必要である。従来技術では２回の丸め演算が必要であった。１つは乗算の後、１つは加算の後で行なわれるものである。これら２回の丸め演算で、精度が失われることがある。たとえば、ｍ＝８を使うと、

$a = 0.11111110 \times 2^0$

$b = 0.10000001 \times 2^1$

$c = -0.1 \times 2^1$の場合、

$a \times b = 0.11111111111111 0 \times 2^0$

（８桁で丸めると）$= 0.1 \times 2^1$

$a \times b + c = 0.1 \times 2^1 - 0.1 \times 2^1$

$= 0$

１回の演算を行なう場合は、

乗算の全精度が加算を通じて保持されるので、

$a \times b + c = -0.00000000000001 \times 2^0$

$= -0.1 \times 2^{-13}$

組み合わせた乗算機構と加算機構の入力ポート及び出力ポートの数は、３つの入力ポートと１つの出力ポート、すなわち４ポートであることに留

意されたい。これは、乗算機構と加算機構がいずれも２つの入力ポートと１つの出力ポート、すなわち合計６つのポートを有する従来技術よりも著しく少ない。したがって、４アドレス・フィールドをもつ単一の命令が、組合せ乗算加算機構にアドレスすることができ、浮動小数点数演算用の命令の長さが著しく減少する。

パイプライン式レジスタを、全加算機構１８と増分機構２０の前に挿入すると好都合である。乗算と２つのオペランドへの簡約の遅延は加算の遅延と同程度なので、パイプラインの各段が都合よくバランスがとれる。さらに、ラッチしなければならないビット数は、大体４ｍ（乗算）＋ｍ（オーバーフロー）であり、したがってパイプライン段の効率が上がる。

部分乗算機構１２として使用できる一部の乗算ツリーは、Ｃ shifted を遅延なしに乗算に挿入できるようにする追加入力を有する（第６図）。ただし、最悪の場合のペナルティは、繰上げ／保管加算機構からのもので、サイクル・タイム中の

わずか数パーセントである。このため、乗算を加算と組み合わせても、乗算の速度にわずかな影響しか及ばない。

部分乗算機構１４は、上記のように、互いに加え合わせると所望の結果に等しくなる２つの部分積をもたらす。このような乗算機構を構成する方法は多数あるが、本発明の好ましい実施例では、ウォーレス・ツリーと呼ばれている構造を使って、かなり速い演算を実現する。

ウォーレス・ツリーの動作を理解するには、まず、第３図に示すようなアレイ・マルチプレクサの動作を理解するのが有用である。説明の都合上、２個の４ビット数を掛け合わせるのに適合した４ビットのアレイ乗算機構を示す。本発明のほとんどの実施例では、ずっと多数のビットに作用することになる。この説明では、第３図の乗算機構は、数$A_1A_2A_3A_4$と$B_1B_2B_3B_4$を掛け合わせる場合について示す。ただし、Ａｉ及びＢｉは、それぞれ４ビット数Ａ及びＢの各ビットを表わす。

第３図の乗算機構は、複数のセル、５０～５３、

７０～７３、９０～９１、１１０～１１３から構成されている。これらの各セルは、それぞれＡＮＤゲート５４～５７、７４～７７、９４～９７、１１０～１１７を含む。各ＡＮＤゲートの入力は、それぞれ、掛け合わそうとする特定のＡｉとＢｉに結合され、ＡＮＤゲートは基本的には単一ビット乗算を行なう。このことは、１だけ及び０だけが掛け合わされ、その乗算の結果も１または０にしかならないことを考慮すると、直観的に明らかになる。ＡＮＤゲートはこの機能を提供する。

各ビットは個別に乗じることができるが、個々の乗算の結果を加え合わせることも必要である。各セルはまた、全加算機構６０～６３、８０～８３、１００～１０３、１２０～１２３を含む。これらの全加算機構は３つの入力ポートを有する。このうち２つの入力ポートは加え合わせようとするビット、すなわち多ビット加算機構における前の加算機構からのキャリー・インと次の加算機構のキャリー・インに向かうキャリー・アウトを受け取るためのものである。全加算機構６０～６３

はアレイ中の第１グループなので、その入力の１つは、それぞれ０に設定されている。また、アレイ中の各行の最上位セルのキャリー・アウトは、その下のセルの入力ポートに送られる。この型式の構造は、ある数の各桁に乗数の１桁を掛けるという人間が筆算で行なうのと同じ型式の加算を実行する。乗数中の後続の数の結果は、それぞれ１０進法で１桁ずつ右にシフトされ、次いでシフトされた結果が加えられる。すると出力１３０～１３７が最終結果を有することになる。

このような乗算機構は、数が長い径路をたどるため、速度が遅い。たとえば、セル５３からのキャリー・アウトは、最終結果に達する前に、８個のセル（５３、５２、７３、７２、９３、９２、１１３、１１２）を通過しなければならない。しかし、同様な方式に基づくはるかに速い乗算機構を作成することができる。

速い乗算機構の１つを第４図に示す。この乗算機構は、キャリー・アウトがすぐ下の左斜め下のセルのキャリー・インに供給される点以外は、第

３図に示したものとよく似ている。当業者なら理解できるように、このような構造が許されるのは、依然としてキャリー・アウトが第３図の乗算機構の場合と同じ重みをもつ列に加えられているためである。加算機構６０～６３は、もはやその隣接する加算機構からキャリー・インを受け取らないので、それらのキャリー・インは０に設定される。この乗算機構は、繰上げが同じ長さの経路を横切らなくてよいので、より速くなるのは明白である。たとえば、６３からのキャリー・アウトは、４個の加算機構、６３、８３、１０３、１２３を通過するだけでよい。この構造がもつと思われる２つの欠点は、この構造が最終結果でなく２つの部分積を生成することと、より多くの配線を使用することである。しかし、この２つの部分積は、１６などの繰上げ／保管加算機構によって最終結果に導くことができる。

出力の各リードは部分積を含むが、たとえば、いくつかのリード対、すなわち１４１と１４２、１４３と１４４、１４５と１４６は同じ重みをも

ち、全加算機構によって加え合わされる。その他のリード、すなわち１４０、１４８、１４９、１５０も部分積を含むが、これらのビット位置での部分積はこの構造によって解決ずみである。それらの部分積は、そのままで使用できるが、全加算機構に供給する場合には、全加算機構の入力の１つをゼロに設定する必要がある。この構造は第３図に示したものよりもかなり高速であるが、さらに改良を加えることが可能である。

第５図は、さらに高速の乗算機構を示す。第５図の乗算機構では、全加算機構のキャリー・アウトが単にその斜め下の加算機構にジャンプするのではなく、２行下に（やはり、その直ぐ左隣りの列に）ジャンプする。この構造は、中間結果が通過する距離がさらに短いので、より高速である。出力１６１、１６２、１６３及び１６４、１６５、１６６及び１６７、１６８、１６９は、それぞれ、同じ重みをもち、繰上げ／保管加算機構によって加え合わされて、２つの出力をもたらす。リード１７０、１７１、及び１７２、１７３及び１７４、

１７６も、同じ重みを有する。リード１６０及び１７８は、既に１ビットになっており、したがって、加算機構を追加する必要はない。

第６図は、Ｊ．Ｌ．ベーア（Baer）の著書「コンピュータ・システム・アーキテクチャ（Computer System Architecture）」（メリーランド州ロックヴィル、Computer Science Press、１９８０年刊）のｐｐ．１０８～１１０に記載されている、ウォーレス・ツリーの構成を示す。ウォーレス・ツリーは、基本的に、第５図の構成の拡張である。再び第５図を参照すると、６３などの加算機構は、その入力のうちの２つに０が加えられるだけなので、もはや不必要であることが理解される。多くの行がスキップされる状況では、第６図に示すようなウォーレス・ツリーが得られる。第６図のＡＮＤゲート２００～２１１は、第５図のＡＮＤゲート５０、７１、９２、１１３に対応する。説明の便宜上、第６図は１２ビット乗算方式を示し、第５図は単に４ビット乗算機構である。重要なことであるが、入力２４９は、２２

０、２２２、２２４、２２６への入力よりも３個の繰上げ／保管加算機構の遅延分だけ遅れることが必要である。この入力は、シフト機構が十分に高速であると仮定すると、シフト機構１４及び補数化機構１５からのＣ shifted でよく、繰上げ保管の遅延が追加されずに、乗算加算を行なうことができる。

乗算用配線の複雑さを最小限にするため、繰上げ／保管加算機構よりも強力な構造を用いて、ウォーレス・ツリーをさらに拡張することができる。繰上げ／保管加算機構は、重みが2^0の３つの入力、及び重みが2^1の１つの出力と重みが2^0の１つの出力の２つの出力を有する３／２加算機構（３、２）である。これは、５つの入力／出力接続を有し、入力より出力が１つ少ない。７／３加算機構（７、３）は、重みが2^0の７つの入力、及びそれぞれ重みが2^0、2^1、2^2である３つの出力を有する。この加算機構では出力が入力よりも４つ少ないので、繰上げ／保管加算機構と同じ機能を実行するのに１／４の（７、３）加算機構

しか必要でない。入力と出力の合計数は１０、すなわち繰上げ／保管加算機構の場合の２倍なので、（７、３）加算機構への接続の合計数は、繰上げ／保管加算機構に必要な接続の１／２である。第７Ａ図は繰上げ保管（３、２）加算機構２６０の入出力表現を示し、第７Ｂ図は（７、３）加算機構２７０のそれと同等の入出力表現を示す。

第８図は、Ｃ shifted を入力３２０に加え、シフト及び補数化動作に対して２つの（７、３）加算機構遅延を見込んである、２８ビット乗算ツリーの好ましい実施例を示す。この乗算ツリーは、上記のウォーレス・ツリーと類似しており、７／３加算機構３００～３０６を使用するように拡張されている。入力３２０は、第６図のウォーレス・ツリーの入力２４９に対応し、補数化機構からＣ shifted を受け取る。第６図の場合と同様に、ＡＮＤゲート２９０～２９６が、乗算を行なう。ＡＮＤゲートの構成が、７／３加算機構３０１、３０２、３０３それぞれの入力で反復される。上記のベーアの著書「コンピュータ・システム・アーキテクチャ」のｐｐ．１０８～１１０に述べられているようなブース・コード化を、ＡＮＤゲート２９０～２９６の場所で使用して、入力数を２８×２まで増加させることができる。

Ｆ．発明の効果

本発明によれば、Ａ×Ｂ＋Ｃ型の浮動小数点演算を実行するに際しての、必要なハードウェア、遅延、及び丸め誤差が減少するという優れた効果が得られる。

４．図面の簡単な説明

第１図は、本発明の構成図である。

第２図は、本発明で用いる補数化機構の説明図である。

第３図、第４図、及び第５図は、本発明を説明するのに有用なアレイ乗算機構の説明図である。

第６図は、本発明で部分乗算機構として使用されるウォーレス・ツリーの説明図である。

第７Ａ図は、本発明の部分乗算機構で使用され、（３、２）加算機構と記載される、繰上げ／保管加算機構の説明図である。

第７Ｂ図は、（７、３）加算機構の説明図である。

第８図は、本発明で部分乗算機構に使用される（７、３）加算機構の概略図である。

出願人　インターナショナル・ビジネス・マシーンズ・コーポレーション

代理人　弁理士　頓宮孝一

（外１名）

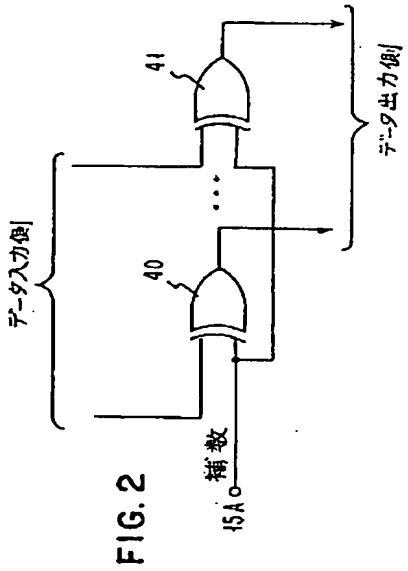

データ入力側
41
40
データ出力側
補数
15A

FIG. 2

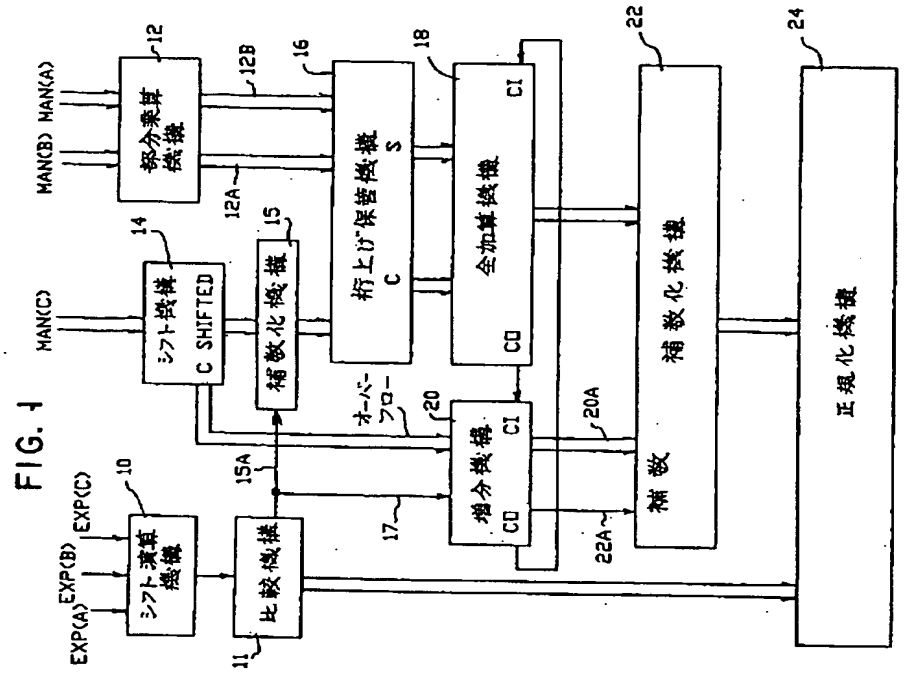

MAN(B)
MAN(A)
MAN(C)
EXP(C)
EXP(B)
EXP(A)
部分乗算機構
シフト機構
C SHIFTED
補数化機構
桁上げ保留機構
S
C
全加算機構
CI
CO
シフト演算機構
比較機構
増分機構
補数化機構
補数
正規化機構
オーバーフロー
12
12A
12B
14
15
15A
16
18
10
11
17
20
20A
22
22A
24

FIG. 1

FIG. 3

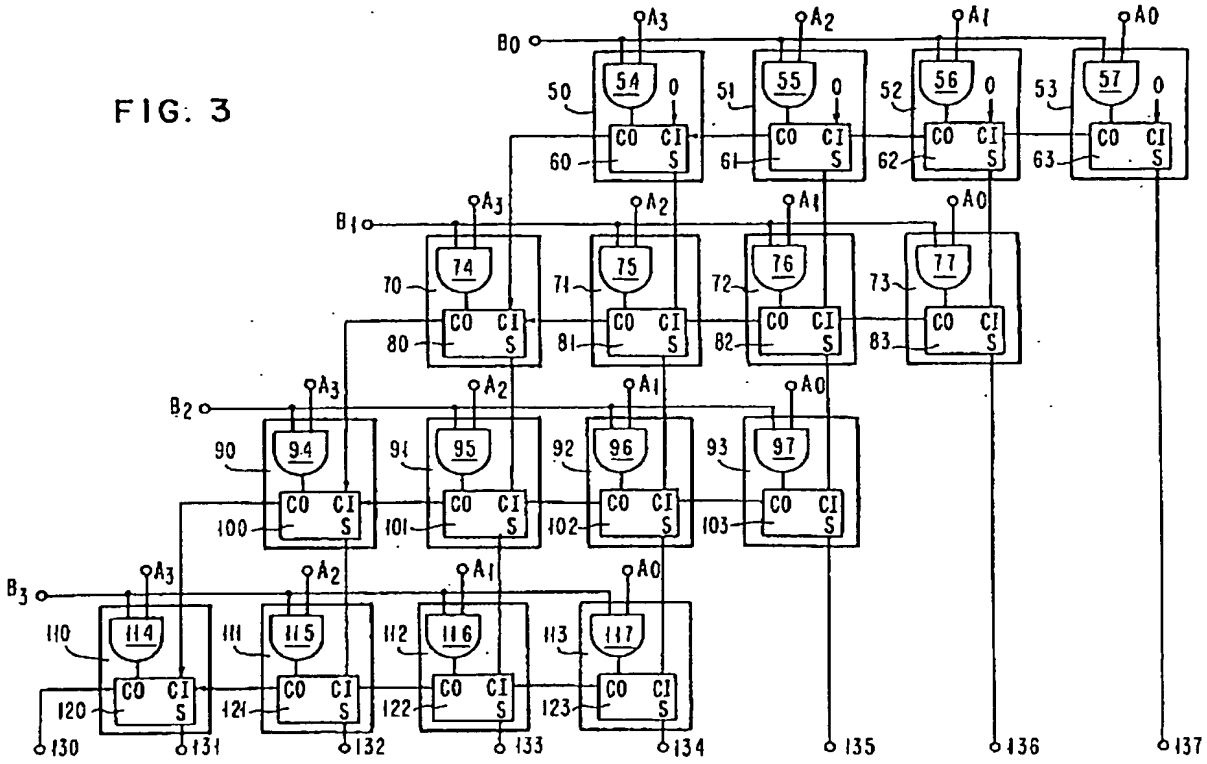

B0
B1
B2
B3
A3
A2
A1
A0
50
51
52
53
54
55
56
57
60
61
62
63
70
71
72
73
74
75
76
77
80
81
82
83
90
91
92
93
94
95
96
97
100
101
102
103
110
111
112
113
114
115
116
117
120
121
122
123
130
131
132
133
134
135
136
137
CO
CI
S
0

FIG. 4

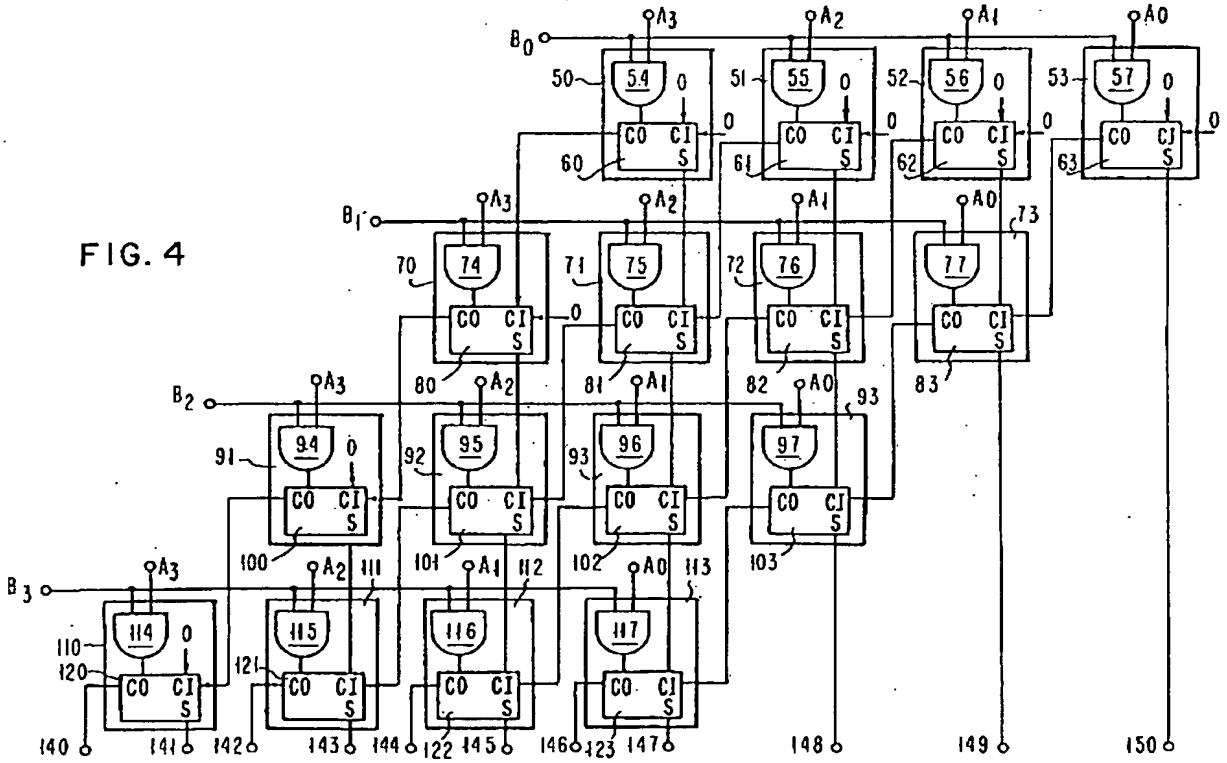
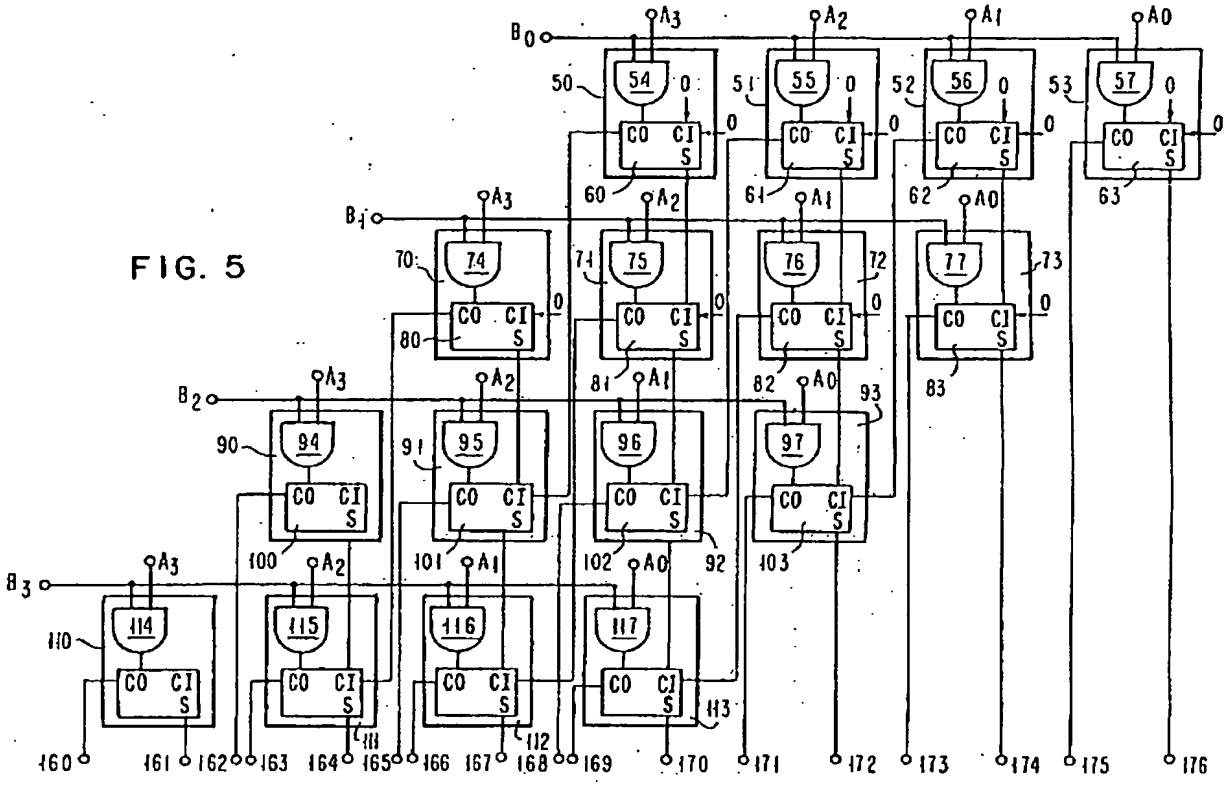

B0
B1
B2
B3
A3
A2
A1
A0
50
51
52
53
54
55
56
57
60
61
62
63
70
71
72
73
74
75
76
77
80
81
82
83
91
92
93
94
95
96
97
100
101
102
103
110
111
112
113
114
115
116
117
120
121
122
123
140
141
142
143
144
145
146
147
148
149
150
CO
CI
S
0

FIG. 5


A3
A2
A1
A0
B0
B1
B2
B3
CO
CI
S
0
50
51
52
53
54
55
56
57
60
61
62
63
70
71
72
73
74
75
76
77
80
81
82
83
90
91
92
93
94
95
96
97
100
101
102
103
110
111
112
113
114
115
116
117
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176

220, 222, 224, 226, 228, 230, 240, 242, 244, 246, 250, 252 …繰上げ/保管/加算機構

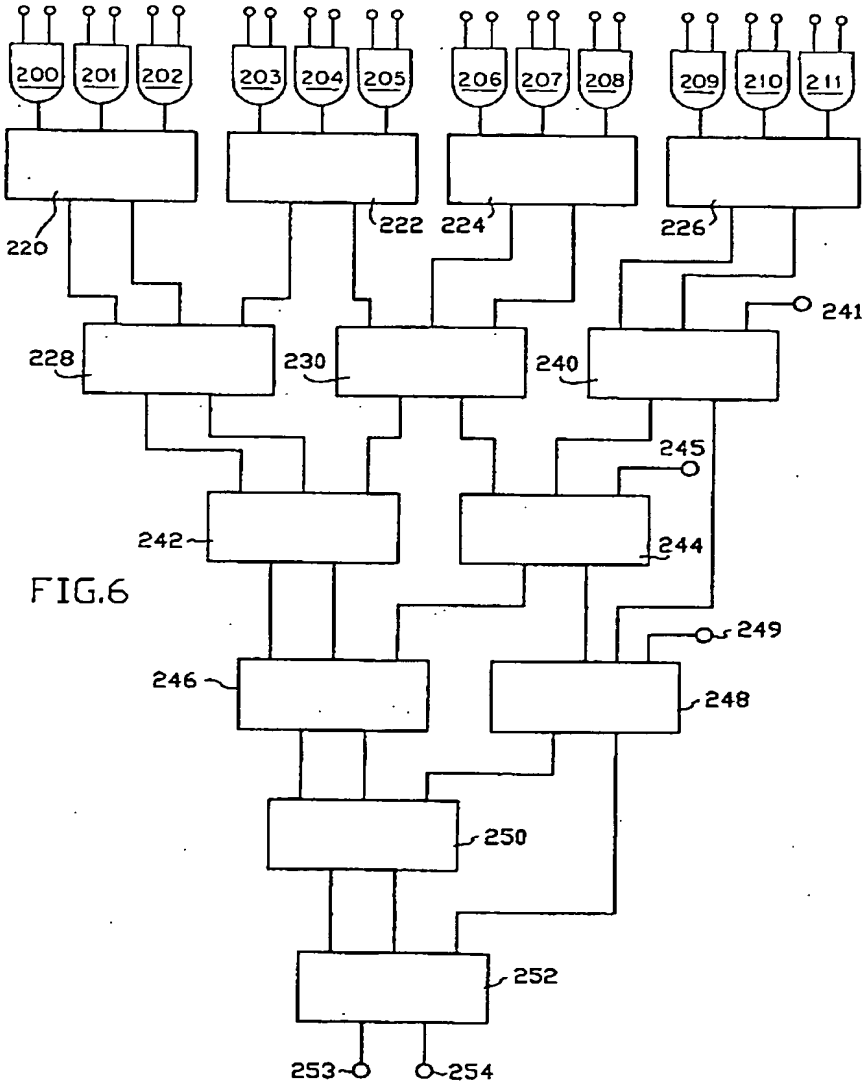
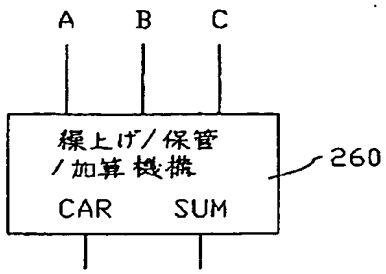


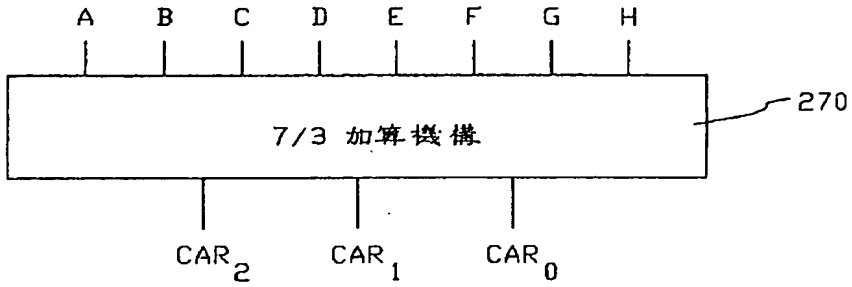
FIG.6

FIG.7A



$$A+B+C+ = CAR \times 2^{1} + SUM \times 2^{0}$$

FIG.7B



$$A+B+C+D+E+F+G = CAR_2 \times 2^{2} + CAR_1 \times 2^{1} + CAR_0$$

FIG. 8